Journal of Computational Physics 523 (2025) 113665

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

FPL-net: A deep learning framework for solving the nonlinear Fokker–Planck–Landau collision operator for anisotropic temperature relaxation

Hyeongjun Noh, Jimin Lee *, Eisung Yoon *

Ulsan National Institute of Science and Technology, Ulsan, 44919, Republic of Korea

ARTICLE INFO

Keywords:
Fokker–Planck collision
Temperature relaxation
Deep learning
Convolutional neural networks
Plasma
Nuclear fusion

ABSTRACT

The nonlinear collision operator consumes a significant amount of computation time in tokamak whole-volume modeling, and in current numerical methods, the computational time grows $O(n^2)$, with n representing the number of plasma species. In this study, we address the acceleration of the Fokker–Planck–Landau (FPL) collision operator using deep learning techniques. The developed FPL-net, a deep learning-based nonlinear Fokker–Planck–Landau collision operator, is a fully convolutional neural network optimized for computational speed with a compact model structure. FPL-net was trained on data representing various temperature conditions of an electron plasma on a two-dimensional velocity grid, ensuring generality. The network's training incorporated physics-informed loss functions for density, momentum, and energy moments of the plasma probability distribution function, which served as constraints, and it was trained to recursively predict two time steps, achieving robust accuracy. Notably, FPL-net demonstrated full temperature relaxation, representing the first time this has been accomplished by a deep learning–based FPL collision operator. Additional experiments with noisy inputs and extended rollouts validated the model's accuracy, which also shows over 1000x acceleration compared to traditional finite volume methods. We discuss the achieved acceleration through deep learning techniques and propose potential avenues for further enhancement and refinement in future research.

1. Introduction

As deep learning technologies continue to advance in areas such as computer vision [1], natural language processing [2], and generative models [3], interdisciplinary research utilizing these technologies has emerged. For example, recent research among mathematicians, computer scientists, and engineers has focused on efficiently solving partial differential equations (PDEs) using machine learning and deep learning methods [4–8]. To accurately solve time-dependent PDEs, methods such as PINN, which utilizes PDEs as the loss function [9,10], have been explored, along with approaches that incorporate error correction [11]. Additionally, improved PDE solvers that achieve more accurate solutions for longer rollouts through iterated refinement steps are being studied [12].

* Corresponding authors.
E-mail addresses: jiminlee@unist.ac.kr (J. Lee), esyoon@unist.ac.kr (E.S. Yoon).

https://doi.org/10.1016/j.jcp.2024.113665
Received 15 January 2024; Received in revised form 5 September 2024; Accepted 6 December 2024

Available online 11 December 2024

In tokamak-based nuclear fusion research, the main goal is to develop and maintain stable discharge scenarios. One recent proposal has suggested using real-time feedback, specifically impurity seeding, to control the detachment of plasma from the material surfaces of the reactor walls [13]. Furthermore, research applying deep learning techniques to the fusion domain is being actively pursued, including scenario development for achieving the desired normalized beta (β_N) using reinforcement learning (RL) [14], optimization of plasma operation using RL [15], magnetic coil control using RL [16], magnetohydrodynamic equilibrium reconstruction [17–19], and disruption forecasting in tokamaks [20]. In addition to optimizing tokamak operation, efforts are also underway to accelerate computationally expensive kinetic simulations [21] such as by encompassing the kinetic effects in comparatively cheap fluid simulations, i.e., closure problems [22–24], using deep learning techniques.

Another computationally demanding problem in nuclear fusion simulations is to solve the nonlinear Fokker–Planck equation for plasma collisions. The mathematical formulation of the Fokker–Planck operator can be represented as an integro-differential equation either in Landau form [25] or Rosenbluth form with potentials [26], both of which require sophisticated numerical approaches. As the exa-scale computing era emerges, whole-device modeling activity requires a nonlinear collision operator to cover the entire region from the core to the wall with one collision model. Correspondingly, continuous research efforts have been made to solve the Fokker–Planck collision operator in both forms, using various numerical methods in a physically reasonable way such that mass, momentum, and energy are conserved [27–32]. Despite such efforts, an inherent disadvantage arises in practice that the calculation speed is slower than linear operators due to the nonlinearity of the collision operator.

To accelerate the computation speed of the Fokker–Planck equation, recent studies have reported the application of deep learning techniques. Chung et al. introduce a vanilla multilayer perceptron (MLP) to compute the drift and diffusion coefficients of the Rosenbluth–Fokker–Planck equation, enhancing computational speed [33]. Miller et al. applied the ReSeg model, developed for semantic segmentation to accelerate the Fokker–Planck–Landau (FPL) collision operator [34]. In the study, they utilized a deep learning method to predict changes in the ion distribution function due to the FPL collision operator. In addition to l_2 loss over the probability distribution function (PDF), loss functions for density, momentum, and energy were defined and used in model training to enforce physical conservation constraints on the neural network. The median relative loss of the model's conservation properties was at the level of 10^{-4}, which would be acceptable for random nature but is not sufficient if it exhibits potential drift nature over time. Therefore, in a subsequent paper [35], the authors proposed a two-loop framework in the training process, where the outer loop optimally updates the weights of physical conservation constraints and the inner loop updates the model parameters by reducing the overall loss, improving the relative error to the level of 10^{-6}. However, as this methodology only handles predictions for a single time step, conducting simulations for thousands of time steps in succession exhibits accumulated time-integrated error, making it difficult to obtain reliable results from long-term simulations.

In this work, we resolve the stability problem by presenting FPL-net, a surrogate FPL collision operator based on deep learning methods, for stable, long rollout simulation of anisotropic electron plasma relaxation. We applied the model structure of U-Net, which is a convolutional neural network, as the backbone of FPL-net [36]. Originally proposed for image segmentation in the biomedical field, U-Net consists of an encoder that captures the context of an input image and a decoder that performs upsampling for precise localization and high resolution. With these characteristics, U-Net demonstrates high performance with a small model size and has been applied in various fields [37]. We chose to utilize U-Net due to its encoder–decoder architecture that maintains the same input and output sizes, preserves high-resolution local information, and achieves strong performance even with a relatively small model size.

FPL-net focuses on fast and stable relaxation simulation. During training, we ensure stable predictions in the presence of perturbations in the input by using the model's output as the subsequent input and computing the output for two future time steps. We implement loss functions for density, momentum, and energy conservation, which are fundamental characteristics of general collision operators, to train the model and ensure that it learns physical laws. As a result, when simulating up to 200 time steps, the time-integrated relative error remained at the level of 10^{-5}. While sustaining such a level of accuracy, our approach demonstrated over 1000 times faster speed compared to traditional numerical analysis algorithms in the tested environment.

In Section 2, we explain the architecture of the convolution-based FPL-net model, which is trained to learn various anisotropic temperature relaxations, and also explain the training and evaluation methods employed to ensure that FPL-net satisfies the physical conservation laws. In Section 3, we present the experimental results of FPL-net. In Section 4, extended experimental results are presented, including long rollouts over the trained time steps and predictions with noisy input, that demonstrate the model's robust stability. In Section 5, we summarize and discuss the performance of FPL-net, clarify the current status, and outline the future work needed.

2. Methods

2.1. Dataset

We prepared training data for FPL-net using a conventional FPL solver that employs a finite volume method with the Picard iteration scheme to solve the integro-differential equation involving the Coulomb collision operator on a two-dimensional velocity grid, which is normalized by the thermal velocity, with a perpendicular axis grid ($N_\perp$) and parallel axis grid ($N_\parallel$) size of $N_\perp \times N_\parallel = 40 \times 60$ [27]. The perpendicular axis ($v_\perp$) denotes the velocity component perpendicular to the magnetic field, and the parallel axis ($v_\parallel$) is the velocity component parallel to the field.

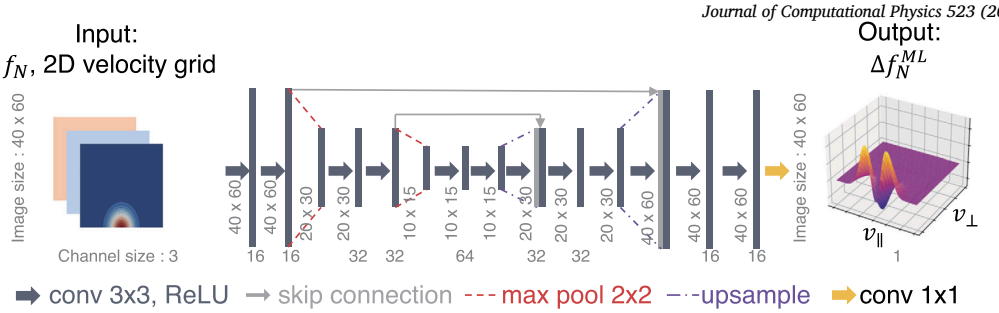


Fig. 1. Schematic of FPL-net for an arbitrary PDF f_N with two-dimensional velocity grid information as inputs. The model is a convolution-based encoder–decoder neural network.

The FPL equation is written in the following form:

$$\frac{df_a}{dt} = \sum_b C_{ab}(f_a; f_b') = -\sum_b \frac{e_a^2 e_b^2 \ln\Lambda_{ab}}{8\pi\epsilon_0^2 m_a} \nabla_v \cdot \int \mathbf{U} \cdot \left(\frac{f_a}{m_b} \nabla_v' f_b' - \frac{f_b'}{m_a} \nabla_v f_a \right) d^3 v', \tag{1}$$

where a and b represent separate species, f_a and f_b denote the PDF of each species, and C_{ab} denotes the collision operator between a and b in respective v and v' coordinates. e, m, ϵ_0, and $\ln\Lambda_{ab}$ are charge, mass, vacuum permittivity, and the Coulomb logarithm respectively. t is time, and $\boldsymbol{U}$ is a tensor defined with relative velocity $\boldsymbol{u} = v - v'$,

$$\boldsymbol{U} = \frac{u^2 \boldsymbol{I} - \boldsymbol{uu}}{u^3}. \tag{2}$$

In the case of three plasma particle species a, b, and c, the change in the PDF of particle a due to collision operations is given by $\frac{df_a}{dt} = C_{aa}(f_a; f_a') + C_{ab}(f_a; f_b') + C_{ac}(f_a; f_c')$. For a single-particle plasma, $\frac{df_a}{dt} = C_{aa}(f_a; f_a')$ and this corresponds to self-collision, which is the current focus.

To ensure the generality of the model, we prepared data for various temperature ratios of bi-Maxwellian cases, covering a broad range of situations. We obtained simulation data from the FPL solver for 115 anisotropic initial temperature cases, with parallel-axis temperature ($T_\parallel$) and perpendicular-axis temperature ($T_\perp$) ratios ranging from $T_\parallel/T_\perp = 0.71$ to 2.19 under a fixed density and perpendicular temperature $T_\perp$ of $n_e = 1.0 \times 10^{19}/m^3$ and $T_\perp = 100$ eV, respectively. A single time step was set to one-tenth of the collision time, and for each of the 115 different anisotropic initial temperature cases, simulations were performed up to 200 time steps. The PDFs at the current and next time steps were paired to form the input and target data for the dataset. To ensure no partial duplication, the dataset was divided based on complete relaxation sequences. For instance, one of the training sets included 200 PDF sequences with $T_\perp/T_\parallel = 0.875$, while the test data included 200 PDF sequences with $T_\perp/T_\parallel = 0.895$. After splitting the data into training, validation, and test sets according to the initial conditions of the relaxation ($T_\perp/T_\parallel$), the data within each of these sets was shuffled and used by the model, thus preventing any overlap between the training and test sets. Of the total 22,885 data, we divided training, validation, and test datasets at a ratio of about 8 : 1 : 1 = train : validation : test.

2.2. Development of the deep learning model, FPL-net

The developed FPL-net is a neural network composed of multiple convolutional layers. The overall encoder–decoder model architecture is shown in Fig. 1. Through the encoder, which reduces the size of the input tensors using max-pooling, the model learns the context of the input. In the decoder, the image size is reconstructed to the original size via upsampling. Between them, feature maps of the encoder are concatenated to the upsampled feature map of the decoder, called skip connections, allowing the model to make use of the localized fine details contained in the encoder. While the architecture of this model generally follows U-Net, the numbers of channels and layers have been optimized to fit our data. An input tensor depth of 3 is formed by concatenating f_N and the grids. Specifically, we stack a PDF at an arbitrary initial condition f_N and two-channel velocity grids, $v_\perp$ and $v_\parallel$, to form a three-channel tensor with the size $40 \times 64 \times 3$. The first channel is f_N, the second channel is $v_\perp$, and the third channel is $v_\parallel$. Both $v_\perp$ and $v_\parallel$ are normalized to the thermal velocity, v_{th}, ensuring that they share the same values across the entire dataset. By stacking a PDF and velocity grid, the model is intended to learn the geometric information of the velocity related to the PDF at each point, allowing it to conserve momentum and energy. As an output, the model computes the PDF change Δf_N^{ML} due to Coulomb collisions at one time step, which is represented as a one-channel tensor of size 40×60. By summing the input f_N and output Δf_N^{ML}, we can predict the next time step PDF f_{N+1}^{ML}. We note that by predicting the PDF change Δf_N^{ML} instead of directly predicting f_{N+1}^{ML}, future applications may consider multiple species, as follows. Considering n species of particles, the change in the PDF of one species can be represented as the sum of collisions with each of the other n species, and therefore predicting Δf_N^{ML} is more extendable for future studies. Moreover, since the value of Δf_N^{ML} is small compared to f_N, learning transformation to Δf_N^{ML} is easier for the model than transformation to f_{N+1}^{ML}, which is almost the same as an identical matrix.

2.3. Physics-informed loss function

One common way to train a deep learning model is by reducing the mean squared error (MSE) loss function using backpropagation. The MSE for n predictions from n observed points is defined as

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2, \tag{3}$$

where Y_i is the observed value and $\hat{Y}_i$ is the predicted value. For our model, the observed value is $Y = f_N$ and the predicted value $\hat{Y}$ corresponds to f_N^{ML}. Moreover, the collision operator must satisfy the conservation properties of density, momentum, and energy [38], which are described by moments of PDF based on kinetic theory in statistical physics as follows:

$$\begin{aligned} n_N &= \int f_N \, d^3v, \\ P_N &= \int m_e v f_N \, d^3v, \\ E_N &= \int \frac{1}{2} m_e v^2 f_N \, d^3v, \end{aligned} \tag{4}$$

where m_e is electron mass and v is particle velocity from the velocity grid information. Based on these definitions, we use three conservation losses between two sequential time steps N and $N+1$ for FPL-net:

$$\begin{aligned} \mathcal{L}_n(f_{N+1}^{ML}; f_{N+1}) &= \left| \frac{n_{N+1}(f_{N+1}^{ML}) - n_{N+1}(f_{N+1})}{n_{N+1}(f_{N+1})} \right|, \\ \mathcal{L}_P(f_{N+1}^{ML}; f_{N+1}) &= \left| \frac{P_{N+1}(f_{N+1}^{ML}) - P_{N+1}(f_{N+1})}{n m_e v_{th,e}} \right|, \\ \mathcal{L}_E(f_{N+1}^{ML}; f_{N+1}) &= \left| \frac{E_{N+1}(f_{N+1}^{ML}) - E_{N+1}(f_{N+1})}{E_{N+1}(f_{N+1})} \right|, \end{aligned} \tag{5}$$

where $v_{th,e}$ is the thermal velocity, $v_{th,e} = \sqrt{T_e/m_e}$. By combining Eq. (3) and Eq. (5), we can drive FPL-net to learn the collision operator by reducing MSE and also enforcing conservation properties. These features are enforced by the overall physics-informed loss function:

$$\begin{aligned} \mathcal{L}^{N+1} &= MSE(f_{N+1}^{ML}; f_{N+1}) \\ &+ \lambda[\mathcal{L}_n(f_{N+1}^{ML}; f_{N+1}) + \mathcal{L}_P(f_{N+1}^{ML}; f_{N+1}) + \mathcal{L}_E(f_{N+1}^{ML}; f_{N+1})], \end{aligned} \tag{6}$$

where λ is a variable hyperparameter that represents the weight of the conservation loss in the overall loss function. Initially, this hyperparameter is set to 0 for our model to learn the overall behavior of the collision operator using MSE loss only. As the training proceeds, the weight λ of the conservation loss function is linearly increased to 0.5 by 600 epochs, allowing the model to be fine-tuned to satisfy the conservation laws. After 600 epochs, λ is fixed to 0.5.

2.4. Training process

The network was trained for 1150 epochs, with each epoch representing a full pass through the training set, using a batch size of 300. One goal of model training was to achieve a robust reliability that ensures the lowest possible time-integrated error for sequential simulations. With the aforementioned loss function only, the model easily predicted Δf_N^{ML} from the given f_N, but its prediction for the next time step, Δf_{N+1}^{ML}, from the input $f_N + \Delta f_N^{ML}$ was not reliable enough. To ensure that our model converges to equilibrium robustly, we added the input f_N to the first prediction Δf_N^{ML} to form the estimated PDF at the next time step, f_{N+1}^{ML}. Then the model predicted Δf_{N+1}^{ML} and subsequently f_{N+2}^{ML}, and calculated the loss functions between f_{N+2}^{ML} and f_{N+2} as well as between f_{N+1}^{ML} and f_{N+1}. Thus, the final loss to predict two future time steps is the sum of these, $\mathcal{L} = \mathcal{L}^{N+1} + \mathcal{L}^{N+2}$. In our model, as demonstrated by other research, utilizing a loss function based on two future time steps improved the performance of the neural network in continuous time-series simulation [39]. This updating process is shown in Fig. 2.

In the training process, it was assumed that each time step depends only on the previous time step, following a Markov chain. The dataset was randomly shuffled and divided into batches during training, and the MSE defined in Eq. (3) was computed by taking the average value over each batch. Similarly, the three conservation losses given in Eq. (5) were also averaged over each batch. In preprocessing, f_N was normalized to the range [0,1], and to ensure that the change in bi-Maxwellian PDF was symmetric to the perpendicular axis ($v_\perp$), we randomly flipped the data on $v_\perp$ with a 30% probability. The model used the Adam optimizer [40], and in each layer, we applied Kaiming He initialization and batch normalization, as well as the ReLU activation function, except for the last 1x1 convolutional layer as shown in Fig. 2, which was used to reduce the number of channels. The model has 95,873 trainable parameters. We implemented a learning rate decay strategy using an exponential decay function with a decay rate

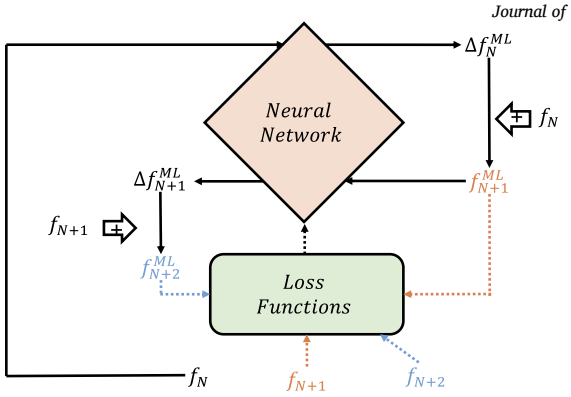


Fig. 2. Overview of the training process. Starting from input f_N, we first estimate Δf_N^{ML} and add it to f_N to produce f_{N+1}^{ML}. Then the neural network takes f_{N+1}^{ML} as an input to produce the next time simulation. By comparing f_{N+1}^{ML} to f_{N+1} and f_{N+2}^{ML} to f_{N+2}, we get two losses for predicting two future time steps, and the network is updated based on their average loss.

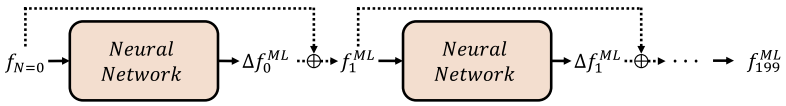


Fig. 3. Flowchart of the model evaluation process. The dotted lines represent the process of adding the model's input and output for use as the input for the next time step. This process is repeated to evaluate the model under 11 different temperature ratios ranging from $T_\parallel/T_\perp = 0.745$ to $T_\parallel/T_\perp = 1.98$ within the test dataset.

of 0.95. This strategy gradually reduced the learning rate from an initial value of $lr = 5.0\text{e}{-4}$ to a final value of $lr = 1.04\text{e}{-18}$. To prevent overfitting, we employed a validation mechanism where the model was assessed using an independent validation dataset after each epoch. The model was updated only if the validation metric improved. In our experiments, we set a termination criterion that halted the training process if the model remained unchanged for an extended period of time, specifically exceeding 427 epochs. All experiments were implemented using PyTorch and accelerated by a single NVIDIA A5000 GPU. With this configuration, the training process took 36 h.

2.5. Evaluation of FPL-net

For an evaluation metric, we used peak signal-to-noise ratio (PSNR):

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{s^2}{\text{MSE}}\right), \tag{7}$$

where s is the maximum value of the data and MSE is defined as in Eq. (5). In the training process, the input data was shuffled randomly, and the model received the PDF of a random initial temperature condition and a random time step, meaning that FPL-net can handle any shape of PDF, and it predicted two future time steps. To evaluate the trained FPL-net, we focus on whether the given PDF converges to equilibrium ($T_\perp = T_\parallel$) via collisions of sequential FPL-net operations. To check for convergence, we take only the initial PDF $f_{N=0}$ from the test dataset and simulate it over 200 sequential time steps. We check whether it converges to equilibrium without accumulating errors until the last time step. This evaluation process is explained in Fig. 3. The initial PDF is given as the first input $f_{N=0}$, and then the network predicts Δf_0^{ML} and adds it to $f_{N=0}$ to get f_1^{ML}. This process is repeated until we obtain f_{199}^{ML} to reach equilibrium. The figure shows the evaluation of a single anisotropic temperature condition, which was repeated 11 times under different conditions. After predicting the anisotropic temperature relaxation using FPL-net for each of the 11 different temperature ratios over time steps f_1^{ML} to f_{199}^{ML}, encompassing a total of 2189 PDFs, the model's predictions were compared with the corresponding target values. We first quantitatively evaluated the model by calculating the overall PSNR between the set of f_N^{ML} and f_N for every test set. Finally, we evaluated the three conservation quantities—density, momentum, and energy—at each prediction.

3. Experimental results

3.1. Performance of FPL-net

Fig. 4 presents the conservation quantities for each test case, evaluated according to the methodology described in Section 2.5. This figure depicts the density, momentum, and energy errors, defined in Eq. (5), for each of the 2189 test cases. Fig. 4(a) illustrates the results obtained by autoregressive prediction up to the 200th time step using the previous model output as input, for each of the 11 anisotropic temperature conditions within the test dataset. For this sequential simulation, the density error exhibited a mean of 9.82×10^{-6} and a median of 7.81×10^{-6}, while the momentum error had a mean of 5.46×10^{-6} and a median of 4.27×10^{-6}. The energy error showed a mean of 2.85×10^{-5} and a median of 2.13×10^{-5}. Fig. 4(b), on the other hand, plots the results obtained

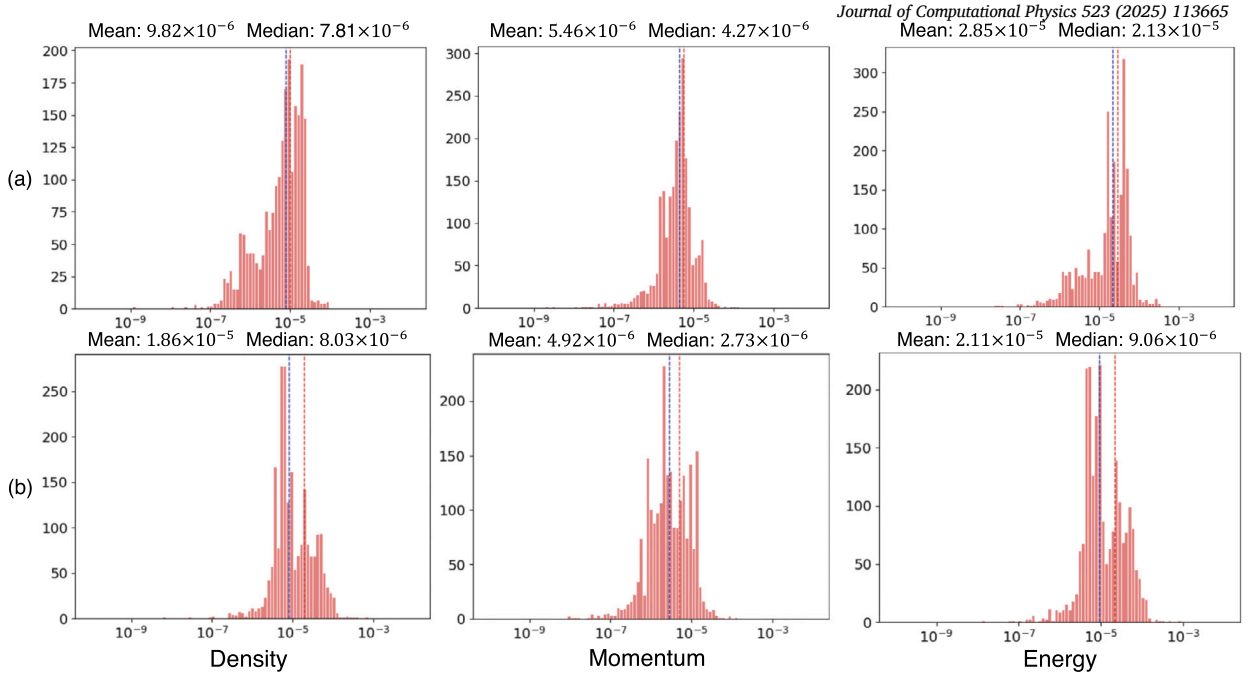


Fig. 4. Histogram of density (left), momentum (middle), and energy (right) error distributions from a total of 2189 test cases. The red vertical dashed line shows the mean value and blue represents the median value of each error. **(a)** Sequential prediction results, and **(b)** single-time-step prediction results. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

by predicting the next single time step in a non-sequential manner, using input f_N from the same test set and repeating this over the whole test dataset. In this case, the density error exhibited a mean of 1.86×10^{-5} and a median of 8.03×10^{-6}, the momentum error had a mean of 4.92×10^{-6} and a median of 2.73×10^{-6}, and the energy error showed a mean of 2.11×10^{-5} and a median of 9.06×10^{-6}. Notably, all errors fall within a tolerable range between 10^{-5} and 10^{-6}, with the largest error being on the order 10^{-5}.

During training, the model predicts Δf_N^{ML} from the input f_N. However, during evaluation, the predictions are made in an autoregressive, sequential manner, where each prediction is based on the previous output. This approach inherently introduces noise or errors between predictions. Despite this, the model consistently demonstrates stable prediction of PDF changes toward equilibrium, as shown by the comparison between sequential [Fig. 4(a)] and non-sequential [Fig. 4(b)] predictions, without the accumulation of errors. Remarkably, the predictions based on the previous output exhibit comparable performance to those obtained by predicting from input f_N, showcasing stable simulation even for multiple sequential time steps. The conservation quantities remain consistent, underscoring the model's ability to avoid error accumulation and maintain robust performance throughout the simulation process. The mean PSNR of the test set is 28.00, and the fact that the error order remains consistent between the two experiments validates the model's performance, covering both single prediction and time-series prediction.

We then conducted a temperature relaxation test on FPL-net, as shown in Fig. 5. The initial conditions were set with a $v_\perp$ temperature of 100 eV and a $v_\parallel$ temperature of 79.5 eV, giving a ratio of $T_\parallel/T_\perp$ of 0.795. Through a comparison with the ground truth data derived from the Picard iteration collision code, we observed that the bi-Maxwellian relaxation exhibited a remarkable agreement, differing by a maximum 3.3% over 199 time steps. This alignment between the results of FPL-net and the ground truth data signifies its accuracy and demonstrates that our model has successfully trained the long-term behavior of the FPL collision operator.

As shown in Fig. 6, we analyzed the resulting errors in density, momentum, and energy over the 199 time steps involving two distinct initial conditions, namely $T_\parallel/T_\perp = 0.795$ and $T_\parallel/T_\perp = 1.98$. The energy error tended to be slightly larger than the others, particularly in the case of $T_\parallel/T_\perp = 1.98$ where the order was around 10^{-4}, while the other errors retained an order of 10^{-5}. Nevertheless, all errors fall within an acceptable range, and only minimal discrepancy was found between the two initial conditions. The error levels remained consistent without any noticeable accumulation of errors. This observation underscores the robustness of our model's results. Regardless of whether the $v_\parallel$ temperature surpasses the $v_\perp$ temperature or vice versa, our model consistently upholds the principles of the physical conservation laws in its results.

Fig. 7 illustrates two test examples with different initial temperature ratios: one with $T_\parallel/T_\perp = 0.795$ predicting Δf_4^{ML}, and the other with $T_\parallel/T_\perp = 1.980$ of the same 4th time step. Fig. 7(a) shows the simulation result of the 4th time step under the initial condition of a temperature ratio between $v_\parallel$ and $v_\perp$ of 0.795. Because the $v_\parallel$ temperature is lower in this case, the simulation exhibits transport to $v_\parallel$ due to collisions. Fig. 7(b) shows the simulation result of the 4th time step under $T_\parallel/T_\perp = 1.980$. In this case, the $v_\perp$ temperature is lower, showing transport in the opposite direction as Fig. 7(a).

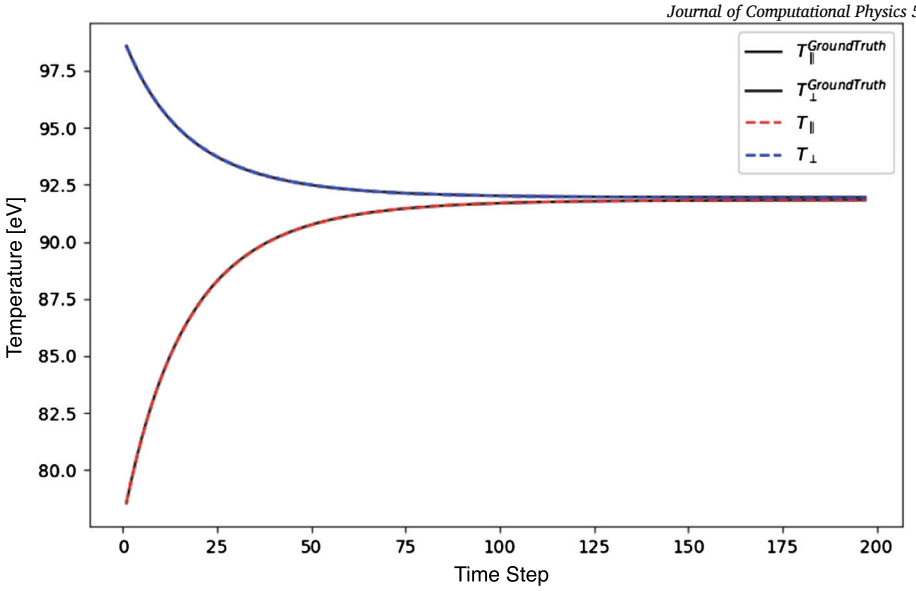


Fig. 5. Outcomes of the temperature relaxation experiment under an initial temperature condition of $T_{\parallel}/T_{\perp} = 0.795$ over 199 time steps. The black solid lines correspond to the ground truth data, while the red (blue) dotted line illustrates the $v_{\parallel}$ ($v_{\perp}$) temperature derived from the results of FPL-net. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

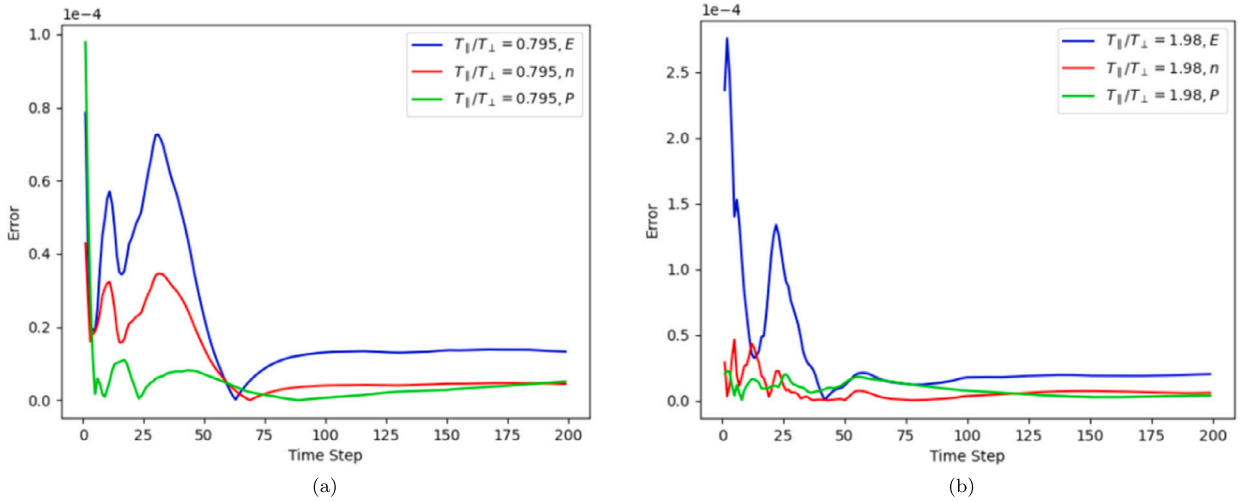


Fig. 6. Visualization depicting the three conservation errors observed over a 199 time step simulation. Energy errors are denoted by blue lines, density errors by red lines, and momentum errors by green lines. (a) Results derived from the initial condition of $T_{\parallel}/T_{\perp} = 0.795$, and (b) results derived from the initial condition of $T_{\parallel}/T_{\perp} = 1.98$. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

3.2. Computational cost of FPL-net compared to numerical methods

While maintaining errors on the order of 10^{-5}, FPL-net also achieved a notable reduction in computation cost compared to traditional numerical methods. Using a single NVIDIA RTX A5000 GPU to measure the inference time and memory usage of the model, FPL-net took an average of 3.56 ms per time step and used 63.82 MB of memory during the inference, whereas the Picard iteration–based collision operator used for data generation took an average of 4135 ms per time step on an Intel Xeon Silver 4112 CPU and used 1017 MB of memory during computation. This difference arises from the fact that the conventional collision operator involves constructing a mesh and iterating until the numerical error falls below a certain threshold, whereas FPL-net loads pretrained model parameters to perform straightforward matrix computations. The advantage of the deep learning approach lies in its ability to provide fast and accurate predictions with minimal computational effort once training is complete.

4. Supplementary experimental results

In this section, we examine the robustness of FPL-net through various additional experiments to determine the extent of its stability.

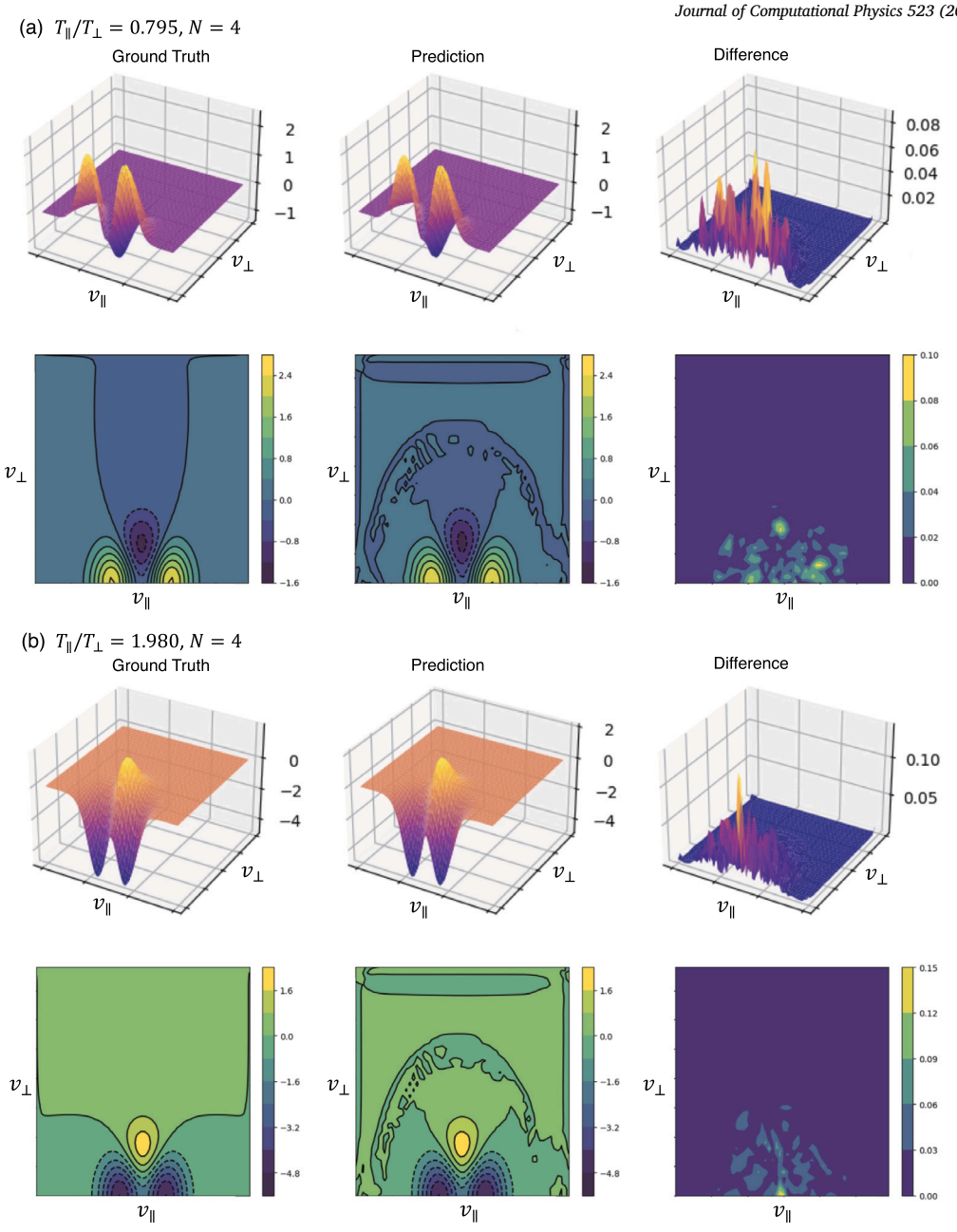


Fig. 7. Experimental results presented for two anisotropic temperature conditions: **(a)** $T_\parallel/T_\perp = 0.795$ and **(b)** $T_\parallel/T_\perp = 1.980$. The figure includes the ground truth from the numerical code (left), FPL-net prediction (middle), and the difference between them (right), along with a three-dimensional plot of the PDF above and its contour map below. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

4.1. Extended rollout

FPL-net was trained for 200 time steps, a duration sufficient to observe temperature relaxation as shown in Fig. 5. But in iterated multi-step forecasting approaches like FPL-net, each prediction relies on the previous one, making them susceptible to error accumulation and potential instability over long rollouts. To assess the boundaries of FPL-net's stability, we extended the prediction length to a maximum of 1200 time steps, which is six times longer than the training length. As shown in Fig. 8, the error remains within an acceptable range and the temperature closely matches the true values up to approximately 1000 time steps. However, beyond this point, the error accumulates and eventually diverges thereafter. Although iterated multi-step forecasting, like in our model, cannot entirely avoid error accumulation, the results demonstrate that predictions remain valid up to five times the length of the time step trained, indicating sufficient stability.

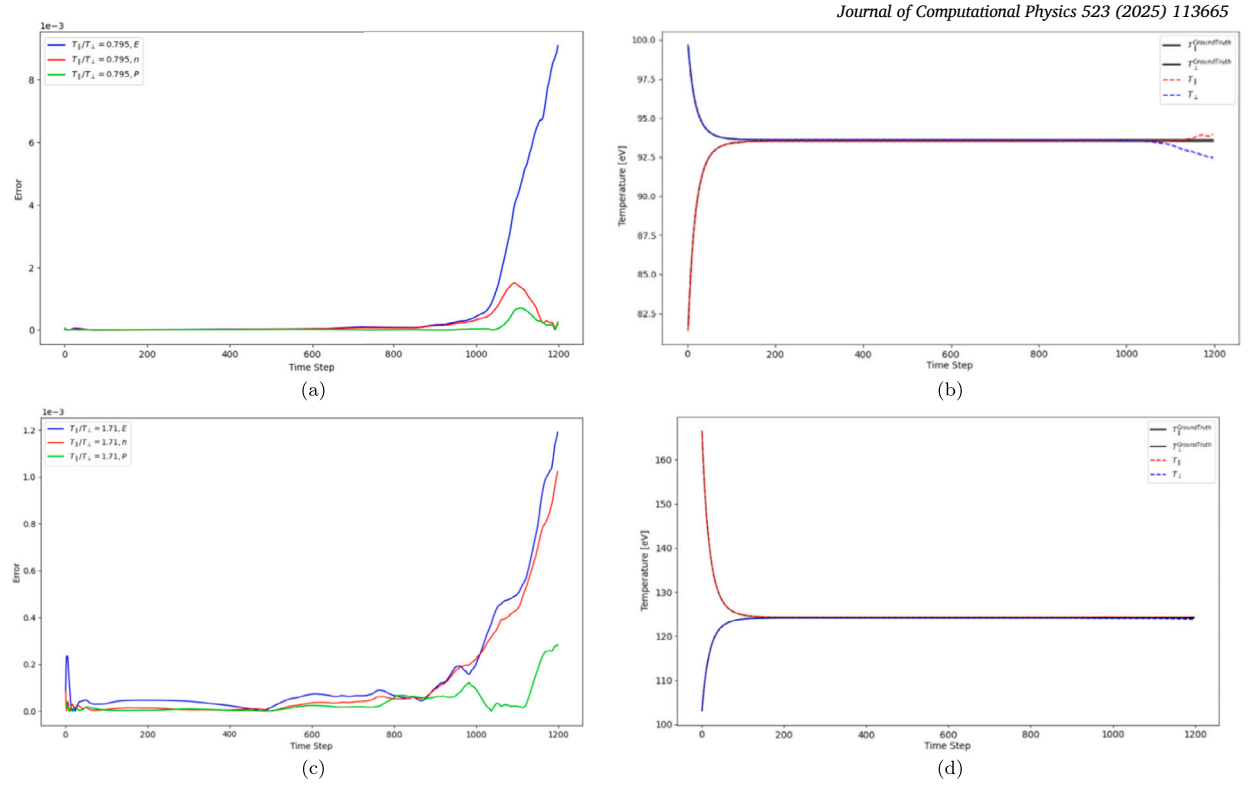


Fig. 8. (a,c) Cumulative error graphs for density (n), parallel momentum (P), and energy (E) over a 1200 time step rollout, with initial conditions of $T_\parallel/T_\perp = 0.795$ (a) and $T_\parallel/T_\perp = 1.71$ (c). (b,d) Comparison between the model predictions (dashed line) and ground truth (solid line) for $T_\parallel$ and $T_\perp$ for the initial conditions of $T_\parallel/T_\perp = 0.795$ (b) and $T_\parallel/T_\perp = 1.71$ (d). (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

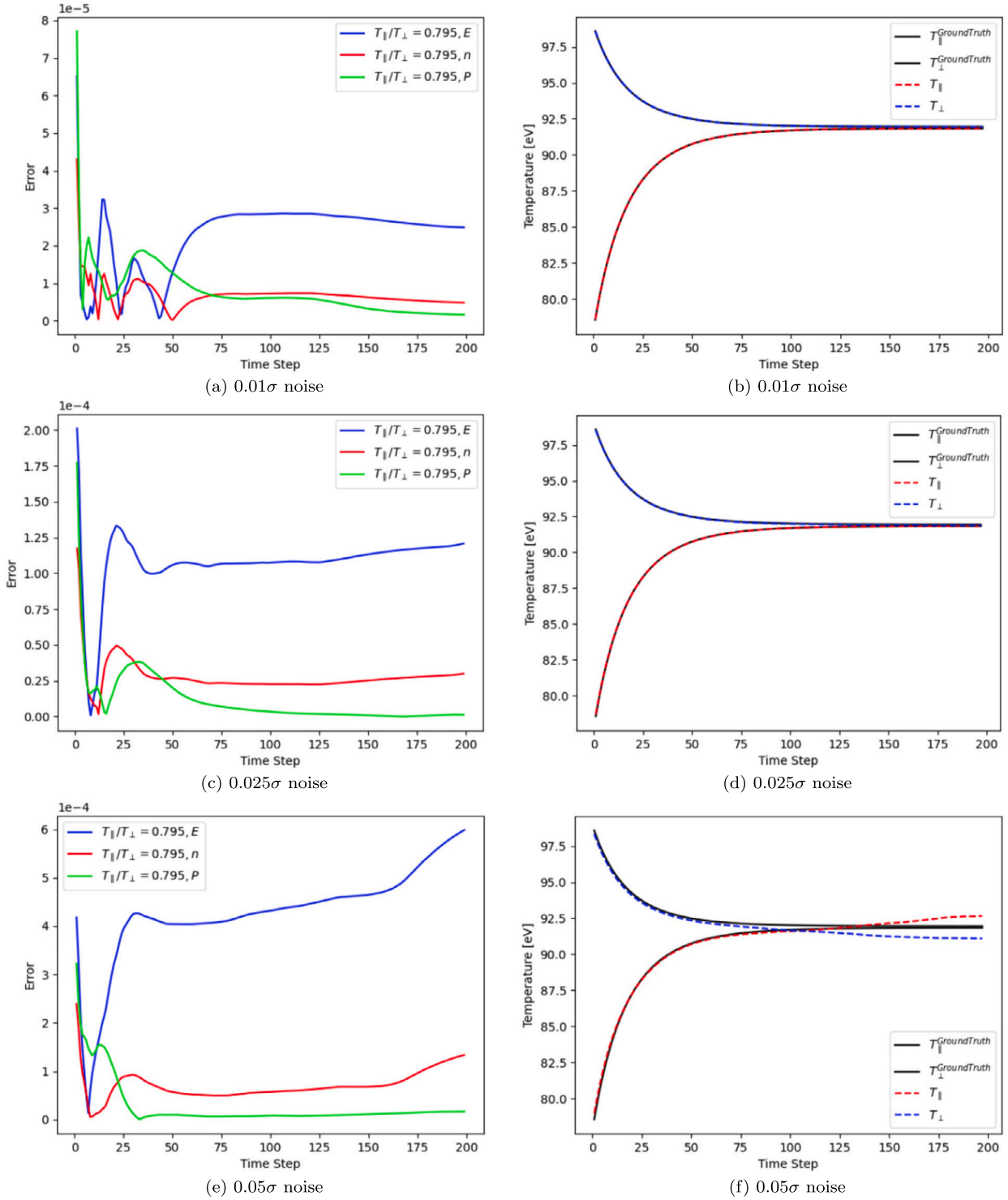
4.2. Addition of Gaussian noise

The data used for training and evaluating FPL-net consists of ideal distribution functions generated by a finite volume method-based code, free from numerical noise. However, for potential applications involving particle-in-cell (PIC) codes, where inherent numerical noise is inevitable, the model must demonstrate robust predictions even with noisy inputs. To assess this, Gaussian noise corresponding to 1%, 2.5%, and 5% of the input's standard deviation (σ) was added, and predictions were conducted up to 200 time steps. As shown in Fig. 9, the results indicate that for noise levels up to 2.5% of σ, the error does not diverge as the time steps progress, and temperature relaxation is successfully achieved. However, when noise at the 5% level is introduced, the model maintains a temperature close to the ground truth only up to approximately 50 time steps, after which the error accumulates progressively, preventing the temperature from reaching equilibrium. Although FPL-net has not been explicitly trained on noisy data, it demonstrates the ability to predict accurately within certain noise thresholds.

5. Discussion and conclusion

We have presented a deep learning–based approach to accelerate the computation of a nonlinear Fokker–Planck–Landau collision operator for fusion plasma. Our proposed FPL-net, composed of convolutional layers, was trained to learn the changes in the probability density functions resulting from self-collisions among electrons under anisotropic temperature conditions sampled from a bi-Maxwellian distribution. To ensure the satisfaction of conservation laws for physical quantities, we incorporated density, momentum, and energy loss functions in the training process.

Our findings revealed that FPL-net maintains numerical stability in the time evolution of the PDFs by utilizing loss functions that consider two sequential time steps, i.e., the previous output is used as the input to predict the next time step. With this feature, FPL-net can achieve convergence to temperature relaxation without accumulating errors. Evaluation results on the test dataset demonstrated that even when simulating up to the relaxation point, FPL-net maintains the same order of error as when predicting a single time step. This study presents, for the first time in this field, the results of testing temperature relaxation using a deep learning–based FPL solver without error accumulation. To further test the robustness of FPL-net, we conducted additional experiments involving very long rollouts and noisy inputs, which were not included in the training data, and found the results to successfully demonstrate temperature relaxation Along with an error tolerance suitable for fusion simulation applications, the developed network achieved over a 1000-fold acceleration in the speed of the nonlinear collision operation compared to previous numerical methods on a CPU.



(a) 0.01σ noise (b) 0.01σ noise

(c) 0.025σ noise (d) 0.025σ noise

(e) 0.05σ noise (f) 0.05σ noise

Fig. 9. FPL-net prediction with random noise. The left column shows the error over 200 time steps for each noise level, and the right column shows the temperature relaxation test results. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

FPL-net serves as a notable example of applying deep learning techniques to plasma collision operators and illustrates a potential surrogate model for numerical methods that can significantly reduce computational speed without sacrificing accuracy. By accelerating computational speeds, it becomes possible to efficiently utilize limited computing resources and facilitate an increased number of simulations within the same time frame, an advancement that can contribute to the acceleration of plasma research. Furthermore, increased computational speed is imperative for the development of digital twin tokamak technology and whole-device modeling. It should be noted, though, that our results are not yet sufficient for immediate application in whole-device modeling or similar complex tasks. Currently, FPL-net can only simulate electron plasma, excluding the main ions, and the initial PDF is assumed to be bi-Maxwellian. For application in the fusion field, a multi-species FPL collision operator that includes main ions and impurities and is trained for various initial PDFs is essential. Thus, as future work, we plan to expand FPL-net by incorporating additional multi-species collision datasets, aiming to broaden its applicability and usage.

CRediT authorship contribution statement

Hyeongjun Noh: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Data curation. **Jimin Lee:** Writing – review & editing, Supervision, Resources, Project administration, Methodology, Funding acquisition, Conceptualization. **Eisung Yoon:** Writing – review & editing, Supervision, Software, Resources, Project administration, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work received support from various sources. We acknowledge the 2023 Research Fund (1.230020.01) of UNIST (Ulsan National Institute of Science & Technology) and the Korea Health Technology R&D Project through the Korea Health Industry Development Institute (KHIDI) funded by the Ministry of Health & Welfare, Republic of Korea (HI21C1161). Additionally, this work was supported by an Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea Ministry of Science and ICT (MSIT) (No. RS-2020-II201336, Artificial Intelligence Graduate School Program (UNIST)). We also appreciate the funding received from the National Research Foundation of Korea (NRF) funded by the MSIT, Republic of Korea (NRF-2021R1F1A1057818 and RS-2022-00155991) and the Korea Institute of Energy Technology Evaluation and Planning and the Ministry of Trade, Industry & Energy (MOTIE) of the Republic of Korea (grant no. 20214000000410).

Data availability

Data will be made available on request.

References

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–90.
[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Adv. Neural Inf. Process. Syst. 27 (2014).
[4] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[5] Z. Li, N.B. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A.M. Stuart, A. Anandkumar, Fourier neural operator for parametric partial differential equations, CoRR, arXiv:2010.08895, 2020.
[6] L. Lu, P. Jin, G.E. Karniadakis, Deeponet: learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators, CoRR, arXiv:1910.03193, 2019.
[7] Y. Bar-Sinai, S. Hoyer, J. Hickey, M.P. Brenner, Learning data-driven discretizations for partial differential equations, Proc. Natl. Acad. Sci. 116 (31) (2019) 15344–15349.
[8] J. Brandstetter, D. Worrall, M. Welling, Message passing neural pde solvers, arXiv:2202.03376, 2023.
[9] J. Seo, I. Kim, H. Nam, Leveraging physics-informed neural computing for transport simulations of nuclear fusion plasmas, Nucl. Eng. Technol. (2024).
[10] J. Seo, Solving real-world optimization tasks using physics-informed neural computing, Sci. Rep. 14 (202) (2024).
[11] N. McGreivy, A. Hakim, Invariant preservation in machine learned pde solvers via error correction, arXiv:2303.16110, 2023.
[12] P. Lippe, B. Veeling, P. Perdikaris, R. Turner, J. Brandstetter, Pde-Refiner: Achieving Accurate Long Rollouts with Neural Pde Solvers, Advances in Neural Information Processing Systems, vol. 36, Curran Associates, Inc., 2023, pp. 67398–67433.
[13] T. Ravensbergen, M. van Berkel, A. Perek, C. Galperti, B. Duval, O. Février, R. van Kampen, F. Felici, J. Lammers, C. Theiler, et al., Real-time feedback control of the impurity emission front in tokamak divertor plasmas, Nat. Commun. 12 (1) (2021) 1105.
[14] J. Seo, Y.-S. Na, B. Kim, C. Lee, M. Park, S. Park, Y. Lee, Feedforward beta control in the kstar tokamak by deep reinforcement learning, Nucl. Fusion 61 (10) (2021) 106010.
[15] T. Wakatsuki, T. Suzuki, N. Hayashi, N. Oyama, S. Ide, Safety factor profile control with reduced central solenoid flux consumption during plasma current ramp-up phase using a reinforcement learning technique, Nucl. Fusion 59 (6) (2019) 066022.

[16] J. Degrave, F. Felici, J. Buchli, M. Neunert, B. Tracey, F. Carpanese, T. Ewalds, R. Hafner, A. Abdolmaleki, D. de Las Casas, et al., Magnetic control of tokamak plasmas through deep reinforcement learning, Nature 602 (7897) (2022) 414–419.
[17] S. Joung, J. Kim, S. Kwak, J. Bak, S. Lee, H. Han, H. Kim, G. Lee, D. Kwon, Y.-C. Ghim, Deep neural network Grad–Shafranov solver constrained with measured magnetic signals, Nucl. Fusion 60 (1) (2019) 016034.
[18] A. Kaptanoglu, M. Landreman, et al., Equilibrium solver using physics-informed neural networks, Bull. Am. Phys. Soc. (2022).
[19] B.P. van Milligen, V. Tribaldos, J. Jiménez, Neural network differential equation and plasma equilibrium solver, Phys. Rev. Lett. 75 (20) (1995) 3594.
[20] J. Kates-Harbeck, A. Svyatkovskiy, W. Tang, Predicting disruptive instabilities in controlled fusion plasmas through deep learning, Nature 568 (7753) (2019) 526–531.
[21] R. Kube, R. Churchill, B. Sturdevant, Machine learning accelerated particle-in-cell plasma simulations, arXiv preprint, arXiv:2110.12444, 2021.
[22] C. Ma, B. Zhu, X.-Q. Xu, W. Wang, Machine learning surrogate models for Landau fluid closure, Phys. Plasmas 27 (4) (2020).
[23] R. Maulik, N.A. Garland, J.W. Burby, X.-Z. Tang, P. Balaprakash, Neural network representability of fully ionized plasma fluid model closures, Phys. Plasmas 27 (7) (2020).
[24] L. Wang, X. Xu, B. Zhu, C. Ma, Y.-a. Lei, Deep learning surrogate model for kinetic Landau-fluid closure with collision, AIP Adv. 10 (7) (2020).
[25] L. Landau, Kinetic equation for the Coulomb effect, Phys. Z. Sowjetunion 10 (1936) 154.
[26] M.N. Rosenbluth, W.M. MacDonald, D.L. Judd, Fokker-Planck equation for an inverse-square force, Phys. Rev. 107 (1) (1957) 1.
[27] E. Yoon, C. Chang, A Fokker-Planck-Landau collision equation solver on two-dimensional velocity grid and its application to particle-in-cell simulation, Phys. Plasmas 21 (3) (2014).
[28] R. Hager, E. Yoon, S. Ku, E.F. D'Azevedo, P.H. Worley, C.-S. Chang, A fully non-linear multi-species Fokker–Planck–Landau collision operator for simulation of fusion plasma, J. Comput. Phys. 315 (2016) 644–660.
[29] W.T. Taitano, L. Chacon, A.N. Simakov, An adaptive, conservative 0d-2v multispecies Rosenbluth–Fokker–Planck solver for arbitrarily disparate mass and temperature regimes, J. Comput. Phys. 318 (2016) 391–420.
[30] D. Kim, J. Seo, G. Jo, J.-M. Kwon, E. Yoon, Nonlinear Fokker-Planck collision operator in Rosenbluth form for gyrokinetic simulations using discontinuous Galerkin method, Comput. Phys. Commun. 279 (2022) 108459.
[31] M. Francisquez, J. Juno, A. Hakim, G.W. Hammett, D.R. Ernst, Improved multispecies Dougherty collisions, J. Plasma Phys. 88 (3) (2022) 905880303.
[32] W. Taitano, L. Chacón, A. Simakov, K. Molvig, A mass, momentum, and energy conserving, fully implicit, scalable algorithm for the multi-dimensional, multi-species Rosenbluth–Fokker–Planck equation, J. Comput. Phys. 297 (2015) 357–380.
[33] K. Chung, F. Fei, M. Gorji, P. Jenny, Data-driven stochastic particle scheme for collisional plasma simulations, J. Comput. Phys. 492 (2023) 112400.
[34] M.A. Miller, R.M. Churchill, A. Dener, C. Chang, T. Munson, R. Hager, Encoder–decoder neural network for solving the nonlinear Fokker–Planck–Landau collision operator in xgc, J. Plasma Phys. 87 (2) (2021) 905870211.
[35] A. Dener, M.A. Miller, R.M. Churchill, T. Munson, C.-S. Chang, Training neural networks under physical constraints using a stochastic augmented Lagrangian approach, arXiv preprint, arXiv:2009.07330, 2020.
[36] O. Ronneberger, P. Fischer, T. Brox, U-net: convolutional networks for biomedical image segmentation, in: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, in: Proceedings, Part III, vol. 18, Springer, 2015, pp. 234–241.
[37] W. Liu, J. Luo, Y. Yang, W. Wang, J. Deng, L. Yu, Automatic lung segmentation in chest x-ray images using improved u-net, Sci. Rep. 12 (1) (2022) 8649.
[38] P. Helander, D.J. Sigmar, Collisional Transport in Magnetized Plasmas, vol. 4, Cambridge University Press, 2005.
[39] B. Ummenhofer, L. Prantl, N. Thuerey, V. Koltun, Lagrangian fluid simulation with continuous convolutions, in: International Conference on Learning Representations, 2019.
[40] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv preprint, arXiv:1412.6980, 2014.